Automated muscle path calibration with gradient-specified optimization based on moment arm

Ziyu Chen[1,2], Tingli Hu[2,3], Sami Haddadin[2,3], David W. Franklin[1,2,4*]

1 Neuromuscular Diagnostics, TUM School of Medicine and Health, Technical University of Munich, Munich, Germany
2 Munich Institute of Robotics and Machine Intelligence (MIRMI), Technical University of Munich, Munich, Germany
3 Chair of Robotics and Systems Intelligence (RSI), TUM School of Computation, Information and Technology, Technical University of Munich, Munich, Germany
4 Munich Data Science Institute (MDSI), Technical University of Munich, Munich, Germany

* david.franklin@tum.de

Abstract

Muscle path calibration is a fundamental step in musculoskeletal modeling, as it determines moment arm and hence the kinetic characteristics of the muscle. However, this task can be laborious, where a large number of path-related parameters must be tuned to match a high-dimensional moment arm–joint angle relation. Here, the process of parameter tuning is formulated as a least-squares problem with a moment arm–based cost function, and this optimization problem is solved with its gradient specified. To derive the gradient analytically, the cost function is first smoothed into a differentiable form by replacing the conditional statements and indifferentiable components with soft functions, and then dissembled into the product of multiple modules, whose gradients are easier to derive in separation. For calibration and validation, a 12-DoF 42-muscle shoulder–arm model is utilized to generate artificial data, and the optimization is configured with non-strict preconditions and constraints. With the specified gradient, the calibration of 42 muscles is completed in 3.7 min, and the validation error is on average 0.12 mm for 182 moment arms. The performance is further compared with two other optimization methods in four conditions. Our method offers a once-and-for-all solution to calibrating the classic obstacle-set path, and the concept employed in gradient derivation is applicable to many other cost functions.

Author summary

Muscle path modeling is more than just routing a cable that visually represents the muscle, but rather it defines how moment arms vary with different joint configurations. The muscle moment arm is the factor that translates muscle force into joint moment, and this property has a huge impact on the accuracy of musculoskeletal simulation. However, it is not easy to calibrate muscle path based on moment arm, because each path is configured by various parameters while the relations between moment arm and both the parameters and joint configuration are complicated. Here, we tackle this challenge in the simple fashion of optimization, but with an emphasis on the gradient; when specified in its analytical form, optimization speed and accuracy will be greatly

improved. We explain in detail how to differentiate the enormous cost function and how our optimization is configured, then we demonstrate the performance of this method by fast and accurate replication of muscle paths from a state-of-the-art shoulder–arm model. As long as the muscle is represented as a cable wrapping around obstacles, our method could free the trouble of path calibration, both for developing generic models and for customizing subject-specific models.

Introduction

In musculoskeletal modeling, the geometry of a muscle is often represented by a series of straight and curved cables, namely the muscle path. It is defined by the locations of the origin, via, and insertion points as well as the size(s), location(s), and orientation(s) of the obstacle(s) around which the cables wrap [1–4]. Although geometrically simplified, muscle path can still have similar muscle length–joint angle and moment arm–joint angle relations to its anatomical reference, provided it is well configured [5–8]. These two relations are crucial to simulation accuracy since muscle length is a major variable in the contraction dynamics [9–12] and moment arm directly determines the kinetic capacity of a muscle [13–15].

The calibration of muscle path is not only a fundamental step for the development of a generic musculoskeletal model, but also necessary in subject-specific applications [16–20]. Scaling of skeletal geometry, for example, is a common practice of model individualization, where the sizes and/or shapes of the subject's segments are replicated in the model based on anthropometric measurements or inverse kinematics [2, 21]. However, such a scaled model is not yet subject-specific, because the individual characteristics of muscle length and moment arm are only reflected in musculoskeletal geometry, which is different from skeletal geometry [2, 12]. In fact, the scaling of skeletal geometry might distort the biomechanical characteristics which were calibrated to be generically correct. For example, depending on how the path of the triceps surae is defined, enlarging the tibia might result in the Achilles tendon moment arm being increased, unchanged, or even decreased. Thus, for a subject-specific model, a rework of path calibration is necessary after scaling to reflect the individual characteristics of muscle length and moment arm, or at least assure that they remain similar to those in the generic model.

Muscle path calibration is performed based on experimental data such as geometric coordinates from medical images [16–18] or moment arm measurements [6, 22], and this process can be laborious. To begin with, there is a large number of path-related parameters to be tuned. Each muscle path is defined by at least six parameters, including the two 3-D coordinates of the origin and insertion points. In addition, there will be three extra parameters for each via point and many more when obstacles are included to recreate the anatomical feature of muscle geometry [5, 17]. For example, cylinders are often used as obstacles in OpenSim models to prevent the muscle from penetrating into the joint, where the cylinder size, orientation, and location would respectively need one, two, and three parameters to define. In general, monoarticular muscles require between six to 12 parameters, whereas complicated multiarticular muscles may require up to 30 parameters [17].

On top of the number of parameters to be tuned, the sensitivity of their chosen objectives to each parameter is often unclear [20]. Hence, manual tuning tends to be puzzling: e.g., enlarging the obstacle or shifting it away from the center of rotation might both increase the corresponding moment arm, but the effect may differ when the joint configuration changes. Even with optimization, the tuning process is more likely to stuck in local minimum due to the absence of any knowledge regarding the precise sensitivity.

The second challenge in muscle path calibration lies in the high dimensional relation between moment arm and joint configuration. Each muscle, even monoarticular ones, can actuate multiple degrees of freedom (DoF), and each DoF corresponds to one moment arm [23–25]. For instance, the gastrocnemius is a biarticular muscle crossing the knee and ankle, which is typically considered to contain at least three DoFs (ankle plantar-/dorsiflexion, ankle eversion/inversion, knee extension/flexion). Therefore, besides the well-studied Achilles tendon moment arm in the sagittal plane [26–29], it has an extra ankle moment arm [30, 31] and a moment arm about the knee [23, 32]. With all DoFs considered, path calibration involves matching multiple moment arm–joint angle relations altogether, and it is common to run into trouble where the calibration of one moment arm can distort others that are previously calibrated.

Furthermore, each moment arm is affected by all actuating DoFs [30, 33, 34]. Suppose the knee moves while immobilizing the ankle, in theory the Achilles tendon ankle moment arm might still change despite no ankle motion. With this, the moment arm–joint angle relation is no longer depictable by a curve or a surface, but rather requires a cube or even a hypercube to demonstrate. For example, the soleus has two moment arms around the ankle, and trying to calibrate them is similar to matching two pairs of surfaces, which is still viable. While for the three moment arms of the gastrocnemius, the task becomes matching the spatial color for three pairs of cubes. This starts to be hard to imagine, but it is still possible if the cubes are somehow matched slice by slice. But with one more moment arm, e.g. for the rectus femoris or many of the shoulder muscles, the number of dimensions to be matched is beyond three and no longer conceivable, and this forces manual tuning to be simplified, compensating the overall simulation accuracy.

In light of these challenges, we developed a gradient-based method for automated muscle path calibration. Regardless of how many DoFs a muscle actuates, predefined path parameters will be optimized to recreate the given moment arm–joint angle relations. With the gradient specified, this method is advantageous in terms of optimization speed and accuracy.

Materials and methods

Optimization method

The process of parameter tuning is formulated as a least-squares problem with the following cost function

$$J(\boldsymbol{p}) = \sum_{n=1}^{N} \left\| \boldsymbol{r}_{\text{target}}(\boldsymbol{q}_n) - \boldsymbol{r}_{\text{model}}(\boldsymbol{q}_n, \boldsymbol{p}) \right\|_2^{\,2} \to \min, \tag{1}$$

where $\boldsymbol{p}$ denotes muscle path parameters, and $\boldsymbol{r}(\boldsymbol{q})$ is the moment arm at joint configuration $\boldsymbol{q}$. Note that $\boldsymbol{r}$ and $\boldsymbol{q}$ are vectors with the dimension of DoF number, whereas N is the size of the dataset. The target value and the model output are indicated by the subscripts, but to make the following equations concise, the model output will be referred to as $\boldsymbol{r}(\boldsymbol{q}, \boldsymbol{p})$, $\boldsymbol{r}(\cdot, \boldsymbol{p})$, or $\boldsymbol{r}$.

In this study, we limited the obstacle type to cylinder to simplify the discussion, but the same principle applies to other types such as cube and stub described in [1]. Currently, our optimization method requires to decide in advance the number of cylinders for each muscle, and the composition of $\boldsymbol{p}$ varies across muscles with three typical forms:

1. No obstacle: $({}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}}, {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{S}})$

2. Single cylinder: $({}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}}, {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{S}}, R, {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{C}}, \alpha, \beta)$

3. Double cylinders: $({}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}}, {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{S}}, R_1, {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{C}_1}, \alpha_1, \beta_1, R_2, {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{C}_2}, \alpha_2, \beta_2)$

where ${}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}}, {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{S}} \in \mathbb{R}^3$ are the coordinates of the origin and insertion points expressed in their respective joint frames, written as column vectors; e.g., ${}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}} = [{}^{\mathrm{j}}x_{\mathrm{P}} \ {}^{\mathrm{j}}y_{\mathrm{P}} \ {}^{\mathrm{j}}z_{\mathrm{P}}]^{\mathrm{T}}$. The radius of a cylinder is denoted by R (the sign indicates wrapping direction) and the center by ${}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{C}}$, whereas the orientation is defined by two Euler angles α and β. In the following discussion, the origin and insertion points are denoted as P and S, and when wrapped by the cylinder (whose center is denoted as C), the two wrapping points are denoted as Q and T (Fig. 1). Also, the frame in which coordinates are expressed are indicated by the superscript: with `j` for the joint frame, `c` for the cylinder frame, and `w` for the world frame.

Muscle path is configured by $\boldsymbol{p}$ using the obstacle-set method [1]. To reduce computational load, we first took a geometric approach to compute $\boldsymbol{r}(\boldsymbol{q}, \boldsymbol{p})$, which is also based on the principle of virtual work [15] but algorithmically less complex by computing partial velocity terms using the Kane's method [35, 36]. Then, since the typical algorithms solving Eq 1 (e.g., the Trust-region method) require the gradient $\partial J / \partial \boldsymbol{p}$, and by the chain rule

$$\frac{\partial J}{\partial \boldsymbol{p}} = \frac{\partial J}{\partial \boldsymbol{r}} \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{p}} = \sum_{n=1}^{N} \left(2\Big(\boldsymbol{r}(\boldsymbol{q}_n, \boldsymbol{p}) - \boldsymbol{r}_{\mathrm{target}}(\boldsymbol{q}_n) \Big) \right) \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{p}}, \tag{2}$$

we need to specify the gradient of $\boldsymbol{r}(\cdot, \boldsymbol{p})$ in its analytical form. For this, we began with revising the obstacle-set method into a smooth form, such that $\partial \boldsymbol{r} / \partial \boldsymbol{p}$ exists for all $\boldsymbol{p}$. This is accomplished by replacing the conditional statements in the obstacle-set method as well as other indifferentiable components with soft functions which yield almost the same outputs but are continuously differentiable (Fig. 1).

Direct derivation of the gradient is overwhelming considering the complexity of $\boldsymbol{r}(\cdot, \boldsymbol{p})$. It would require a tremendous amount of effort, and the eventual result would be filled with pages of repeated terms. Thus, we circumvented this problem by dissembling it into a composite of multiple gradients whose analytical forms are easier to derive.

Soft functions

Fig 1 (middle) shows the detailed workflow of the revised obstacle-set method, containing two conditional statements. The first statement checks if either of the designated fixed points is within the cylinder. Note that this is not included in the original method [1] (Fig 1, left), and we introduce it to simplify the optimization since it removes the complex geometric constraint between the cylinder and the points. The second statement checks if the wrapping angle exceeds 180°, which is considered invalid wrapping.

Naturally, conditional statements make $\boldsymbol{r}(\cdot, \boldsymbol{p})$ indifferentiable in certain domains, and we introduced soft functions to patch them (Fig 1, right). Step 1.5 essentially keeps a minimum value of R of for the distance between the point and the cylindrical axis (e.g., $d_{\mathrm{P}} = {}^{\mathrm{c}}x_{\mathrm{P}}{}^2 + {}^{\mathrm{c}}y_{\mathrm{P}}{}^2$), which is equivalent to a rectified linear unit (ReLU) activation function. This function is not differentiable when the variable is zero, and we replaced it using a soft variant (Fig 2, top center)

$$f_{\mathrm{softReLU}}(\delta) = \frac{\delta + \sqrt{\delta^2 + (2 + \sqrt{2})a^2}}{2}, \tag{3}$$

where $\delta = d^2 - R^2$, and the value of a determines how steep the transition is.

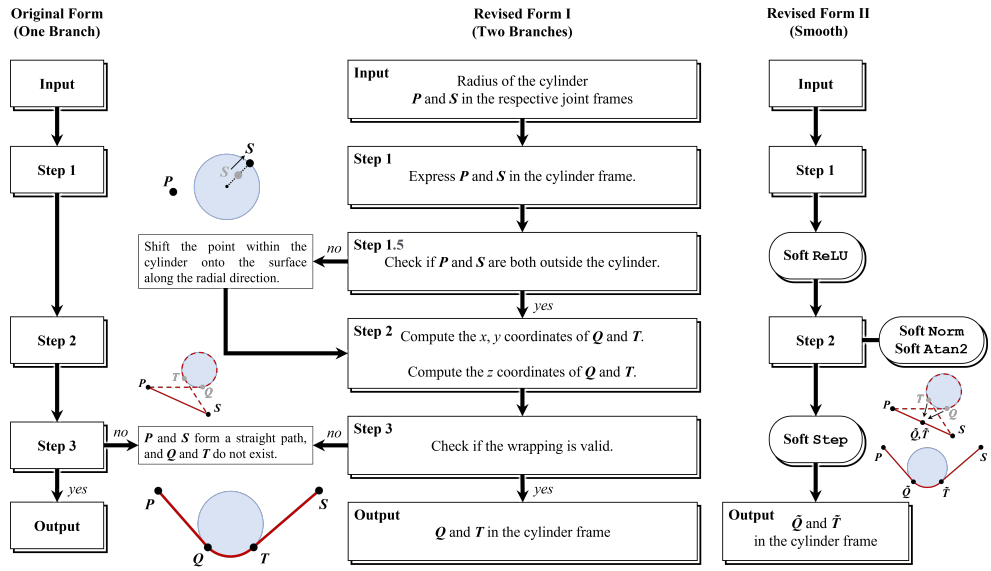


Fig 1. Workflows of the original and revised obstacle-set methods. Left: the original form with one conditional statement. Middle: the revised form with an extra conditional statement for continuity. Right: the smooth form revised for optimization. The naming and order of the origin, wrapping, and insertion points are indicated by the illustration in the bottom.

Eq 3 is derived based on a hyperbola (Fig 2, top left)

$$\frac{x^2}{a^2} - \frac{y^2}{\tan^2(\frac{3\pi}{8})a^2} = 1, \tag{4}$$

whose asymptotes are $y = \pm\tan(\frac{3\pi}{8})x$. When this hyperbola is rotated clockwise by $\frac{3\pi}{8}$, the asymptotes become $y = 0$ and $y = x$, and the left branch becomes the curve described by Eq 3. It is now clear that the geometric meaning of $|a|$ is the distance from the vertices of the hyperbola to its center. Therefore, the smaller it is, the steeper the hyperbola is. In this study, we have $a = 0.01$, and the output of $f_{\mathrm{softReLU}}(x)$ is always slightly greater than 0 when $x < 0$, and is slightly greater than x when $x > 0$.

Although the soft plus function $y = \log(1 + e^x)$ is more typical as a smooth version of ReLU activation function (Fig 2, bottom right), its output may easily exceed the largest floating-point number representable on a computer (Fig 2, bottom center). When $x = 710$, e^x is already greater than the limit of 1.7977×10^{308} in IEEE double precision, and the optimization may risk being terminated if the coordinates of anatomical points are expressed in `mm` and unbounded. Moreover, there is about an error of 31% to 69% when $x \in [-1, 1]$ (Fig 2, top center), so the approximation will not be good when larger units such as `cm` or `m` are used to avoid the aforementioned problem. Whereas for Eq 3, its error around zero can be made sufficiently small with a smaller a (Fig 2, top right), and its output will not risk returning `Inf` in computation.

In Step 3, based on the sign of

$$s = R({}^{\mathrm{c}}x_{\mathrm{Q}}{}^{\mathrm{c}}y_{\mathrm{T}} - {}^{\mathrm{c}}x_{\mathrm{T}}{}^{\mathrm{c}}x_{\mathrm{Q}}), \tag{5}$$

the method returns either the coordinates of the two wrapping points or null if wrapping is invalid (Fig 1, left). The latter situation can be problematic for optimization as it leads to yet another conditional statement in moment arm computation, in which

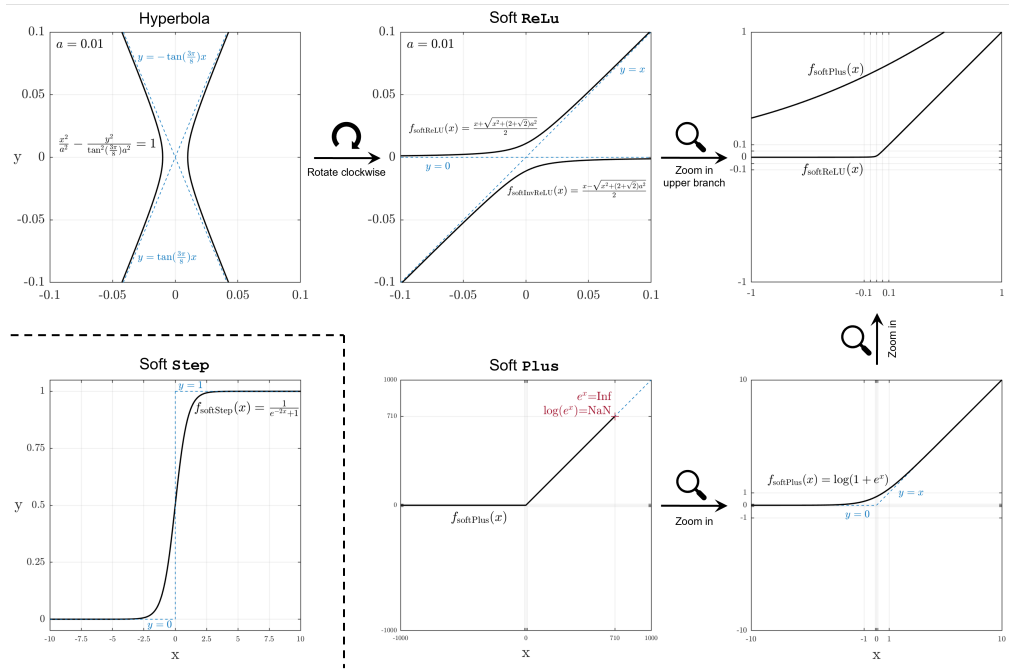


Fig 2. Graphs of soft functions. Top left: hyperbola from which soft `ReLU` is derived. Top center: soft `ReLU` and soft inverse `ReLU`. Top right: soft `ReLU` and soft `Plus`. Bottom left: soft `Step`. Bottom center: soft `Plus` and its undefined area due to MATLAB precision limit. Bottom right: soft `Plus`. Notice the difference in axis magnitude.

different formulas are used depending on whether the wrapping occurs:

$$\text{straight path: } \boldsymbol{r} = \left(\frac{\partial\, {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{P}}}{\partial \boldsymbol{q}} - \frac{\partial\, {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{S}}}{\partial \boldsymbol{q}}\right)^{\mathrm{T}} \frac{{}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{S}} - {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{P}}}{\|\, {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{S}} - {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{P}}\|_2}, \tag{6}$$

$$\begin{aligned}\text{wrapped path: } \boldsymbol{r} = &\left(\frac{\partial\, {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{P}}}{\partial \boldsymbol{q}} - \frac{\partial\, {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{Q}}}{\partial \boldsymbol{q}}\right)^{\mathrm{T}} \frac{{}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{Q}} - {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{P}}}{\|\, {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{Q}} - {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{P}}\|_2} \\ &+\left(\frac{\partial\, {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{T}}}{\partial \boldsymbol{q}} - \frac{\partial\, {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{S}}}{\partial \boldsymbol{q}}\right)^{\mathrm{T}} \frac{{}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{S}} - {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{T}}}{\|\, {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{S}} - {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{T}}\|_2}.\end{aligned} \tag{7}$$

Therefore, the revision of Step 3 into a smooth form involves a transition from Eq 6 to Eq 7 when s changes from positive to negative. Here, ${}^{\mathrm{w}}\boldsymbol{u}$ is transformed from ${}^{\mathrm{j}}\boldsymbol{u}$ or ${}^{\mathrm{c}}\boldsymbol{u}$:

$$\begin{bmatrix} {}^{\mathrm{w}}\boldsymbol{u} \\ 1 \end{bmatrix} = {}^{\mathrm{w}}\boldsymbol{T}_{\mathrm{j}}(\boldsymbol{q}) \begin{bmatrix} {}^{\mathrm{j}}\boldsymbol{u} \\ 1 \end{bmatrix} = {}^{\mathrm{w}}\boldsymbol{T}_{\mathrm{j}}(\boldsymbol{q})\, {}^{\mathrm{j}}\boldsymbol{T}_{\mathrm{c}} \begin{bmatrix} {}^{\mathrm{c}}\boldsymbol{u} \\ 1 \end{bmatrix}, \tag{8}$$

where ${}^{\mathrm{w}}\boldsymbol{T}_{\mathrm{j}}(\boldsymbol{q})$ is the transformation matrix from the joint frame to the world frame, and ${}^{\mathrm{j}}\boldsymbol{T}_{\mathrm{c}}$ cylinder to joint. The complication arises with involvement of the joint frames, indicating the bones on which P, S, and the cylinder are fixed: ${}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}}$ and ${}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{S}}$ must be defined in separate bones, one or both of which are different from the bone for fixating the cylinder. In other words, the implication of `j` in Eq 8 is different between P and Q as well as between T and S, making it hard to cancel out the terms in Eq 7 to transition into Eq 6. To this end, we have Q and T respectively approaching the midpoint between P and S when s tends to 0 in the negative direction, and the coordinates of such transitional points $\tilde{\mathrm{Q}}$ and $\tilde{\mathrm{T}}$ may be calculated using a soft version of Heaviside step function (Fig 2, bottom left):

$$
\begin{aligned}
{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{Q}}} &= \frac{{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}} + {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{S}}}{2} + ({}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{Q}} - \frac{{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}} + {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{S}}}{2}) f_{\mathrm{softStep}}(s), \\
{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{T}}} &= \frac{{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}} + {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{S}}}{2} + ({}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{T}} - \frac{{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}} + {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{S}}}{2}) f_{\mathrm{softStep}}(s),
\end{aligned} \tag{9}
$$

where

$$
f_{\mathrm{softStep}}(s) = \frac{1}{e^{-2s} + 1}. \tag{10}
$$

This way, moment arm computation is generalized as

$$
\begin{aligned}
\boldsymbol{r}(\boldsymbol{q}, {}^{\mathrm{w}}\boldsymbol{u}) = &\Big(\frac{\partial\, {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{P}}}{\partial \boldsymbol{q}} - \frac{\partial\, {}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{Q}}}}{\partial \boldsymbol{q}}\Big)^{\mathrm{T}} \frac{{}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{Q}}} - {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{P}}}{\| {}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{Q}}} - {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{P}} \|_2} \\
&+ \Big(\frac{\partial\, {}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{T}}}}{\partial \boldsymbol{q}} - \frac{\partial\, {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{S}}}{\partial \boldsymbol{q}}\Big)^{\mathrm{T}} \frac{{}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{S}} - {}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{T}}}}{\| {}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{S}} - {}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{T}}} \|_2},
\end{aligned} \tag{11}
$$

When $s > 0$, $\tilde{\mathrm{Q}}$ and $\tilde{\mathrm{T}}$ maintain their original coordinates as wrapping points Q and T (${}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{Q}}} \approx {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{Q}}, {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{T}}} \approx {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{T}}$), and moment arm is computed with Eq 7. When $s < 0$, ${}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{Q}}} = {}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{T}}} \approx ({}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}} + {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{S}})/2$, and Eq 11 simplifies into Eq 6. The wrapping indicator s can be magnified to steepen the transition around 0. Importantly, this does not interfere with the moment arm computation involving two obstacles.

Furthermore, Step 2 involves calculation of the 2-norm and the 2-argument arctangent. Both are indifferentiable at the origin, and we replaced them with

$$
f_{\mathrm{softNorm}}(\boldsymbol{x}) = \sqrt{\|\boldsymbol{x}\|_2{}^2 + n} - \sqrt{n}, \tag{12}
$$

where n should be a sufficiently small positive number (`eps` in this study), and

$$
f_{\mathrm{softAtan2}}(x, y) = 2\arctan\left(\frac{y}{\sqrt{x^2 + y^2} + x}\right), \tag{13}
$$

where x is assigned a negative value with a sufficiently small magnitude (-0.001 in this study), if x is negative but greater than this value. This prevents the denominator term in Eq 13 from becoming 0 and hence avoid discontinuity.

Gradients

With Eqs 3 and 10, the obstacle-set method, or more precisely, the computation of $\boldsymbol{u}(\boldsymbol{q}, \boldsymbol{p})$ is revised into a smooth form, and since Eqs 3, 10, 12, and 13 are clearly all differentiable, it is now possible to compute the gradient $\partial\boldsymbol{u}/\partial\boldsymbol{p}$.

Our target function $\boldsymbol{r}(\boldsymbol{q}, \boldsymbol{p})$ is a composite of $\boldsymbol{u}(\boldsymbol{q}, \boldsymbol{p})$ and the generalized moment arm computation $\boldsymbol{r}(\boldsymbol{q}, \boldsymbol{u})$ with Eq 11, and by the chain rule, we may now have

$$
\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{p}} = \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}} \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{p}}, \tag{14}
$$

which simplifies the derivation by disassembling a complex function into the product of less complicated ones. However at this stage, Eq. 14 is still difficult to compute due to the knotty structure of $\partial\boldsymbol{u}/\partial\boldsymbol{p}$, so we continue down the chain.

Note that in this study, the gradient of a scalar variable is formatted as a row vector, while that of a vector variable is formatted as a matrix, namely the Jacobian, whose rows are the gradients of each scalar variable. Suppose for the single-cylinder case with 12 parameters, ${}^{\partial R}/_{\partial \boldsymbol{p}}$ is simply a 12-D row vector whose seventh element is 1, while the rest are 0, and ${}^{\partial\, \mathrm{j}}\boldsymbol{u}_{\mathrm{P}}/_{\partial \boldsymbol{p}}$ is a 3×3 identity matrix concatenated with a 3×9 null matrix.

In Step 1, the coordinates of the origin and insertion points are transformed from the joint frame to the cylinder frame

$$\begin{bmatrix} {}^{\mathrm{c}}\boldsymbol{u} \\ 1 \end{bmatrix} = {}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j}}({}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{C}}, \alpha, \beta) \begin{bmatrix} {}^{\mathrm{j}}\boldsymbol{u} \\ 1 \end{bmatrix}, \tag{15}$$

and ${}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j}}({}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{C}}, \alpha, \beta)$ is a 4×4 transformation matrix, from the joint frame to the cylinder frame, determined by the location and orientation of the cylinder

$${}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j}} = {}^{\mathrm{j}}\boldsymbol{T}_{\mathrm{c}}{}^{-1} = \left[\begin{array}{ccc:c} \cos\beta & 0 & \sin\beta & {}^{\mathrm{j}}x_{\mathrm{C}} \\ \sin\alpha\sin\beta & \cos\alpha & -\cos\beta\sin\alpha & {}^{\mathrm{j}}y_{\mathrm{C}} \\ -\cos\alpha\sin\beta & \sin\alpha & \cos\alpha\cos\beta & {}^{\mathrm{j}}z_{\mathrm{C}} \\ \hdashline 0 & 0 & 0 & 1 \end{array}\right]^{-1}. \tag{16}$$

Based on the product rule, we may then easily compute the gradient (with the origin point as an example)

$$\begin{aligned} \underbrace{\begin{bmatrix} \frac{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}}}{\partial \boldsymbol{p}} \\ 0 \ldots 0 \end{bmatrix}}_{4\times 12} &= \left[\begin{array}{c:c:c:c} \underbrace{\frac{\partial\, {}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j}}}{\partial\, {}^{\mathrm{j}}x_{\mathrm{P}}}\begin{bmatrix} {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}} \\ 1 \end{bmatrix} \cdots \frac{\partial\, {}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j}}}{\partial\, {}^{\mathrm{j}}z_{\mathrm{S}}}\begin{bmatrix} {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}} \\ 1 \end{bmatrix}}_{4\times 6} & \underbrace{\frac{\partial\, {}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j}}}{\partial R}\begin{bmatrix} {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}} \\ 1 \end{bmatrix}}_{4\times 1} & \underbrace{\frac{\partial\, {}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j}}}{\partial\, {}^{\mathrm{j}}x_{\mathrm{C}}}\begin{bmatrix} {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}} \\ 1 \end{bmatrix} \cdots}_{4\times 3} & \underbrace{\frac{\partial\, {}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j}}}{\partial \alpha}\begin{bmatrix} {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}} \\ 1 \end{bmatrix} \cdots}_{4\times 2} \end{array}\right] \\ &\quad + {}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j}} \underbrace{\begin{bmatrix} \frac{\partial\, {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}}}{\partial \boldsymbol{p}} \\ 0 \ldots 0 \end{bmatrix}}_{4\times 12} \\ &= \left[\begin{array}{c:c:c:c} \underset{4\times 6}{\mathbf{O}} & \underset{4\times 1}{\mathbf{O}} & \underbrace{\frac{\partial\, {}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j}}}{\partial\, {}^{\mathrm{j}}x_{\mathrm{C}}}\begin{bmatrix} {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}} \\ 1 \end{bmatrix} \quad \frac{\partial\, {}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j}}}{\partial\, {}^{\mathrm{j}}y_{\mathrm{C}}}\begin{bmatrix} {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}} \\ 1 \end{bmatrix} \quad \frac{\partial\, {}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j}}}{\partial\, {}^{\mathrm{j}}z_{\mathrm{C}}}\begin{bmatrix} {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}} \\ 1 \end{bmatrix}}_{4\times 3} & \underbrace{\frac{\partial\, {}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j}}}{\partial \alpha}\begin{bmatrix} {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}} \\ 1 \end{bmatrix} \quad \frac{\partial\, {}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j}}}{\partial \beta}\begin{bmatrix} {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}} \\ 1 \end{bmatrix}}_{4\times 2} \end{array}\right] \\ &\quad + {}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{j}} \begin{bmatrix} \underset{3\times 3}{\mathbf{I}} & \underset{3\times 9}{\mathbf{O}} \\ 0 \ldots & 0 \end{bmatrix}, \end{aligned} \tag{17}$$

and notice the partial derivatives with respect to $\boldsymbol{p}$ are in the order of ${}^{\mathrm{j}}x_{\mathrm{P}}$, ${}^{\mathrm{j}}y_{\mathrm{P}}$, ${}^{\mathrm{j}}z_{\mathrm{P}}$, ${}^{\mathrm{j}}x_{\mathrm{S}}$, ${}^{\mathrm{j}}y_{\mathrm{S}}$, ${}^{\mathrm{j}}z_{\mathrm{S}}$, R, ${}^{\mathrm{j}}x_{\mathrm{C}}$, ${}^{\mathrm{j}}y_{\mathrm{C}}$, ${}^{\mathrm{j}}z_{\mathrm{C}}$, α, and β.

Step 1.5 checks if the origin and insertion points P and S are both outside the cylinder, and we utilized Eq 3 to return a positive value $\tilde{\delta}$ and its gradient (with the origin point as an example)

$$\tilde{\delta_{\mathrm{P}}} = f_{\mathrm{softReLU}}({}^{\mathrm{c}}x_{\mathrm{P}}{}^{2} + {}^{\mathrm{c}}y_{\mathrm{P}}{}^{2} - R^2) \tag{18}$$

$$\begin{aligned} \frac{\partial \tilde{\delta_{\mathrm{P}}}}{\partial \boldsymbol{p}} &= \frac{\partial f_{\mathrm{softReLu}}}{\partial \delta} \left[\ \underbrace{\frac{\partial \delta}{\partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}}}}_{1\times 3} \ \vdots\ \frac{\partial \delta}{\partial R}\ \right] \underbrace{\begin{bmatrix} \partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}}/_{\partial \boldsymbol{p}} \\ \hdashline \partial R/_{\partial \boldsymbol{p}} \end{bmatrix}}_{4\times 12} \\ &= -\left(\tfrac{1}{2} + \frac{{}^{\mathrm{c}}x_{\mathrm{P}}{}^{2} + {}^{\mathrm{c}}y_{\mathrm{P}}{}^{2} - R^2}{\sqrt{({}^{\mathrm{c}}x_{\mathrm{P}}{}^{2} + {}^{\mathrm{c}}y_{\mathrm{P}}{}^{2} - R^2)^2 + (2+\sqrt{2})a^2}}\right) \ 2\ \left[{}^{\mathrm{c}}x_{\mathrm{P}}\ \ {}^{\mathrm{c}}y_{\mathrm{P}}\ \ 0 \ \vdots\ -R\right] \begin{bmatrix} \partial\, {}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}}/_{\partial \boldsymbol{p}} \\ \hdashline \underset{1\times 6}{\mathbf{O}}\ \ 1\ \ \underset{1\times 5}{\mathbf{O}} \end{bmatrix}, \end{aligned} \tag{19}$$

where ${}^{\partial\,\mathrm{c}}\boldsymbol{u}_{\mathrm{P}}/\partial\boldsymbol{p}$ is the only variable unspecified in $\boldsymbol{p}$ and may be obtained from Eq 17.

Next, with $\tilde{\delta}_{\mathrm{P}}$ and $\tilde{\delta}_{\mathrm{S}}$ from Eq 18, we can derive the x and y coordinates of the wrapping points Q and T as described in [1]:

$$\begin{aligned}\begin{bmatrix}{}^{\mathrm{c}}x_{\mathrm{Q}}\\{}^{\mathrm{c}}y_{\mathrm{Q}}\end{bmatrix} &= \begin{bmatrix}\frac{{}^{\mathrm{c}}x_{\mathrm{P}}R^2-{}^{\mathrm{c}}y_{\mathrm{P}}R\sqrt{\tilde{\delta}_{\mathrm{P}}}}{{}^{\mathrm{c}}x_{\mathrm{P}}{}^2+{}^{\mathrm{c}}y_{\mathrm{P}}{}^2}\\ \frac{{}^{\mathrm{c}}y_{\mathrm{P}}R^2+{}^{\mathrm{c}}x_{\mathrm{P}}R\sqrt{\tilde{\delta}_{\mathrm{P}}}}{{}^{\mathrm{c}}x_{\mathrm{P}}{}^2+{}^{\mathrm{c}}y_{\mathrm{P}}{}^2}\end{bmatrix},\\ \begin{bmatrix}{}^{\mathrm{c}}x_{\mathrm{T}}\\{}^{\mathrm{c}}y_{\mathrm{T}}\end{bmatrix} &= \begin{bmatrix}\frac{{}^{\mathrm{c}}x_{\mathrm{S}}R^2+{}^{\mathrm{c}}y_{\mathrm{S}}R\sqrt{\tilde{\delta}_{\mathrm{S}}}}{{}^{\mathrm{c}}x_{\mathrm{S}}{}^2+{}^{\mathrm{c}}y_{\mathrm{S}}{}^2}\\ \frac{{}^{\mathrm{c}}y_{\mathrm{S}}R^2-{}^{\mathrm{c}}x_{\mathrm{S}}R\sqrt{\tilde{\delta}_{\mathrm{S}}}}{{}^{\mathrm{c}}x_{\mathrm{S}}{}^2+{}^{\mathrm{c}}y_{\mathrm{S}}{}^2}\end{bmatrix},\end{aligned} \tag{20}$$

and additionally with $\partial\tilde{\delta}_{\mathrm{P}}/\partial\boldsymbol{p}$ and $\partial\tilde{\delta}_{\mathrm{S}}/\partial\boldsymbol{p}$ from Eq 19, we can also compute their gradients

$$\begin{aligned}\frac{\partial\,{}^{\mathrm{c}}x_{\mathrm{Q}}}{\partial\boldsymbol{p}} &= \begin{bmatrix}\frac{\partial\,{}^{\mathrm{c}}x_{\mathrm{Q}}}{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}}} & \frac{\partial\,{}^{\mathrm{c}}x_{\mathrm{Q}}}{\partial R} & \frac{\partial\,{}^{\mathrm{c}}x_{\mathrm{Q}}}{\partial\tilde{\delta}_{\mathrm{P}}}\end{bmatrix}\begin{bmatrix}\frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}}}{\partial\boldsymbol{p}} & \frac{\partial R}{\partial\boldsymbol{p}} & \frac{\partial\tilde{\delta}_{\mathrm{P}}}{\partial\boldsymbol{p}}\end{bmatrix}^{\mathrm{T}},\\ \frac{\partial\,{}^{\mathrm{c}}y_{\mathrm{Q}}}{\partial\boldsymbol{p}} &= \begin{bmatrix}\frac{\partial\,{}^{\mathrm{c}}y_{\mathrm{Q}}}{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}}} & \frac{\partial\,{}^{\mathrm{c}}y_{\mathrm{Q}}}{\partial R} & \frac{\partial\,{}^{\mathrm{c}}y_{\mathrm{Q}}}{\partial\tilde{\delta}_{\mathrm{P}}}\end{bmatrix}\begin{bmatrix}\frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}}}{\partial\boldsymbol{p}} & \frac{\partial R}{\partial\boldsymbol{p}} & \frac{\partial\tilde{\delta}_{\mathrm{P}}}{\partial\boldsymbol{p}}\end{bmatrix}^{\mathrm{T}},\\ \frac{\partial\,{}^{\mathrm{c}}x_{\mathrm{T}}}{\partial\boldsymbol{p}} &= \begin{bmatrix}\frac{\partial\,{}^{\mathrm{c}}x_{\mathrm{T}}}{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{S}}} & \frac{\partial\,{}^{\mathrm{c}}x_{\mathrm{T}}}{\partial R} & \frac{\partial\,{}^{\mathrm{c}}x_{\mathrm{T}}}{\partial\tilde{\delta}_{\mathrm{S}}}\end{bmatrix}\begin{bmatrix}\frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{S}}}{\partial\boldsymbol{p}} & \frac{\partial R}{\partial\boldsymbol{p}} & \frac{\partial\tilde{\delta}_{\mathrm{S}}}{\partial\boldsymbol{p}}\end{bmatrix}^{\mathrm{T}},\\ \frac{\partial\,{}^{\mathrm{c}}y_{\mathrm{T}}}{\partial\boldsymbol{p}} &= \begin{bmatrix}\frac{\partial\,{}^{\mathrm{c}}y_{\mathrm{T}}}{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{S}}} & \frac{\partial\,{}^{\mathrm{c}}y_{\mathrm{T}}}{\partial R} & \frac{\partial\,{}^{\mathrm{c}}y_{\mathrm{T}}}{\partial\tilde{\delta}_{\mathrm{S}}}\end{bmatrix}\begin{bmatrix}\frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{S}}}{\partial\boldsymbol{p}} & \frac{\partial R}{\partial\boldsymbol{p}} & \frac{\partial\tilde{\delta}_{\mathrm{S}}}{\partial\boldsymbol{p}}\end{bmatrix}^{\mathrm{T}},\end{aligned} \tag{21}$$

where the first terms on the right side are easy to derive from Eq 20, so we omit the processes to avoid expanding the results, and the second terms are obtained from Eqs 17 and 19. To be concise, for the rest of this subsection we will only present the formats of the gradients, whose computations share a similar simplicity.

Step 2 ends with the calculation of ${}^{\mathrm{c}}z_{\mathrm{Q}}$ and ${}^{\mathrm{c}}z_{\mathrm{T}}$, and Step 3 returns the coordinates of $\tilde{\mathrm{Q}}$ and $\tilde{\mathrm{T}}$ after checking the validity of the wrapping (Eq 9). Likewise, their gradients are formatted as

$$\begin{aligned}\frac{\partial\,{}^{\mathrm{c}}z_{\mathrm{Q}}}{\partial\boldsymbol{p}} &= \begin{bmatrix}\frac{\partial\,{}^{\mathrm{c}}z_{\mathrm{Q}}}{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}}} & \frac{\partial\,{}^{\mathrm{c}}z_{\mathrm{Q}}}{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{S}}} & \frac{\partial\,{}^{\mathrm{c}}z_{\mathrm{Q}}}{\partial R} & \frac{\partial\,{}^{\mathrm{c}}z_{\mathrm{Q}}}{\partial\,{}^{\mathrm{c}}x_{\mathrm{Q}}} & \frac{\partial\,{}^{\mathrm{c}}z_{\mathrm{Q}}}{\partial\,{}^{\mathrm{c}}y_{\mathrm{Q}}}\end{bmatrix}\begin{bmatrix}\frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}}}{\partial\boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{S}}}{\partial\boldsymbol{p}} & \frac{\partial R}{\partial\boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{c}}x_{\mathrm{Q}}}{\partial\boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{c}}y_{\mathrm{Q}}}{\partial\boldsymbol{p}}\end{bmatrix}^{\mathrm{T}},\\ \frac{\partial\,{}^{\mathrm{c}}z_{\mathrm{T}}}{\partial\boldsymbol{p}} &= \begin{bmatrix}\frac{\partial\,{}^{\mathrm{c}}z_{\mathrm{T}}}{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}}} & \frac{\partial\,{}^{\mathrm{c}}z_{\mathrm{T}}}{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{S}}} & \frac{\partial\,{}^{\mathrm{c}}z_{\mathrm{T}}}{\partial R} & \frac{\partial\,{}^{\mathrm{c}}z_{\mathrm{T}}}{\partial\,{}^{\mathrm{c}}x_{\mathrm{T}}} & \frac{\partial\,{}^{\mathrm{c}}z_{\mathrm{T}}}{\partial\,{}^{\mathrm{c}}y_{\mathrm{T}}}\end{bmatrix}\begin{bmatrix}\frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}}}{\partial\boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{S}}}{\partial\boldsymbol{p}} & \frac{\partial R}{\partial\boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{c}}x_{\mathrm{T}}}{\partial\boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{c}}y_{\mathrm{T}}}{\partial\boldsymbol{p}}\end{bmatrix}^{\mathrm{T}},\end{aligned} \tag{22}$$

$$\begin{aligned}\frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{Q}}}}{\partial\boldsymbol{p}} &= \begin{bmatrix}\frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{Q}}}}{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}}} & \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{Q}}}}{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{S}}} & \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{Q}}}}{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{Q}}} & \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{Q}}}}{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{T}}}\end{bmatrix}\begin{bmatrix}\frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}}}{\partial\boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{S}}}{\partial\boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{Q}}}{\partial\boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{T}}}{\partial\boldsymbol{p}}\end{bmatrix}^{\mathrm{T}},\\ \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{T}}}}{\partial\boldsymbol{p}} &= \begin{bmatrix}\frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{T}}}}{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}}} & \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{T}}}}{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{S}}} & \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{T}}}}{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{Q}}} & \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{T}}}}{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{T}}}\end{bmatrix}\begin{bmatrix}\frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}}}{\partial\boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{S}}}{\partial\boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{Q}}}{\partial\boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{T}}}{\partial\boldsymbol{p}}\end{bmatrix}^{\mathrm{T}},\end{aligned} \tag{23}$$

Here at the end of the obstacle-set method, we have obtained ${}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{P}}(\boldsymbol{p})$, ${}^{\mathrm{c}}\boldsymbol{u}_{\mathrm{S}}(\boldsymbol{p})$, ${}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{Q}}}(\boldsymbol{p})$, and ${}^{\mathrm{c}}\boldsymbol{u}_{\tilde{\mathrm{T}}}(\boldsymbol{p})$ as well as their gradients. Before computing moment arm with Eq 11, we need to utilize Eq 8 to transform them from their respective joint frames to the world frame.

Finally, Eq 14 may now be extended into

$$\frac{\partial\boldsymbol{r}}{\partial\boldsymbol{p}} = \begin{bmatrix}\frac{\partial\boldsymbol{r}}{\partial\,{}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{P}}} & \frac{\partial\boldsymbol{r}}{\partial\,{}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{S}}} & \frac{\partial\boldsymbol{r}}{\partial\,{}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{Q}}}} & \frac{\partial\boldsymbol{r}}{\partial\,{}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{T}}}}\end{bmatrix}\begin{bmatrix}\frac{\partial\,{}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{P}}}{\partial\boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{w}}\boldsymbol{u}_{\mathrm{S}}}{\partial\boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{Q}}}}{\partial\boldsymbol{p}} & \frac{\partial\,{}^{\mathrm{w}}\boldsymbol{u}_{\tilde{\mathrm{T}}}}{\partial\boldsymbol{p}}\end{bmatrix}^{\mathrm{T}}, \tag{24}$$

where the first term on the right side is easily derived from Eq 11, while the second term lies in the chain of Eqs 17, 19, 21, 22, and 23.

Calibration and validation

With the gradient $\partial \boldsymbol{r} / \partial \boldsymbol{p}$ specified, we used the nonlinear least-squares solver (`lsqnonlin`, with `trust-region-reflective` algorithm) in MATLAB to solve Eq 1. A 12-DoF 42-muscle human shoulder–arm model [37] (with muscle path from [17]) was used to compute $\boldsymbol{r}(\boldsymbol{q}, \boldsymbol{p})$, and it was also used as a reference to generate artificial calibration and validation data. Thus in a sense, the algorithm tries to replicate the geometry of all muscle paths in the reference model.

As previously mentioned, each muscle path requires predetermination of the number of via points and cylinders for the structure of $\boldsymbol{p}$. Some muscle paths are modeled with via points and more than two obstacles (S2 Table), and their structures of $\boldsymbol{p}$ would be a combination of the three typical forms. For example, the extensor carpi radialis longus is modeled in the form of

$$\underbrace{\text{origin}\text{—}\overbrace{\text{via}}^{\boldsymbol{p}_{\text{single}}}}_{\boldsymbol{p}_{\text{straight}}}\text{—}\overbrace{\text{cylinder}\text{—}\underbrace{\text{via}}_{\boldsymbol{p}_{\text{straight}}}}^{\boldsymbol{p}_{\text{single}}}\text{—}\overbrace{\underbrace{\text{via}}_{\boldsymbol{p}_{\text{straight}}}\text{—}\text{cylinder}\text{—}\underbrace{\text{via}}_{\boldsymbol{p}_{\text{straight}}}}^{\boldsymbol{p}_{\text{single}}}\text{—}\underbrace{\text{insertion}}_{\boldsymbol{p}_{\text{straight}}},$$

which contains a total of 30 path parameters.

For calibration, we experimented two scenarios. In the first scenario, we set the structure of path parameters same as in the reference model. This is mostly the case of subject-specific modeling, where the parameter structure is known in the generic model and calibration is performed to adjust parameter values. In the second scenario, all muscle paths were configured with either one or two cylinders and without any via points, resulting in a different parameter structure from their reference counterpart (S2 Table), and approximately 10% fewer parameters in total. In other words, the path is calibrated to match its moment arms with those generated by a likely more complicated muscle geometry, representing the case of generic modeling.

Apart from predetermination of parameter structure, two constraints were included in the optimization. Firstly, coordinates of the origin and insertion points (${}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{P}}, {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{S}}$) were bounded within a uniformly distributed range of ± 10 mm from the respective original values in the reference model. This is not a prerequisite for the optimization, but since the origin and insertion points are anatomical landmarks with accurate measurements, we include this constraint to speed up the optimization. Secondly, the sign of the cylinder radius (R) is set the same as in the reference model. We can of course have it tuned in optimization using a soft sign function similar to Eq 10, but this is not necessary because the direction of wrapping can be easily determined from the anatomy.

To reduce the chance of local minimum, the optimization was run with multiple sets of initial parameters using the `MultiStart` solver: For each parameter that a muscle path has (S2 Table), 25 random starting points were generated with the function `CustomStartPointSet`. Additionally, to prevent excess computation, we programmed the optimization so that it will first iterate over 10% of the starting points, and if the error per moment arm does not reach $0.01N$, another 30% will be iterated. The rest of starting points will be iterated only when the calibration error per moment arm fails to reach N after the first 40% of iterations. Consequently, the stopping criteria of the solver can be lenient and were conveniently configured as follows: 10^{-3} for `FunctionTolerance`, `OptimalityTolerance` and `StepTolerance`, and 10^{2} for both `MaxFunctionEvaluations` and `MaxIterations`.

Moreover, when generating the starting points, the coordinates of cylinder center (${}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{C}}$) were randomly initialized around the sections of the line between the initial origin and insertion points. This is an essential step because if the cylinder is not wrapped at the beginning of the optimization, $\partial \boldsymbol{r} / \partial {}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{C}}$ remains a null matrix, and ${}^{\mathrm{j}}\boldsymbol{u}_{\mathrm{C}}$ might be left untuned till the end. Hence, we must ensure that the cylinders are not too departed from the initial muscle path.

We then utilized the kinematic measurements of the shoulder and arm from 10 intransitive daily tasks in [38]: $\boldsymbol{q}_n$ of five tasks were input to the reference model to generate artificial calibration data as $\boldsymbol{r}_{\text{target}}(\boldsymbol{q}_n)$, while the kinematics of another five tasks were used for validation. Due to the sample size of each kinematic measurement, we interpolated the original data to avoid redundant computation: the number of joint configurations for calibration equals to the number of path parameters that each muscle has (S2 Table), and 25 configurations (five instants per movement) were extracted for validation.

To evaluate the contribution of the analytical form of the gradient, the optimization was also performed using `lsqnonlin` without analytical specification, where the gradient is computed numerically. Furthermore, to assess the performance of gradient-based optimization, we included the method of `patternsearch`, which does not require any gradient to solve the optimization problem. Finally, to examine the robustness of our method against error, calibration was also performed with noisy data. For this, the aforementioned artificial moment arm data were added with both relative error (uniformly distributed within ±20% of the reference value) and absolute error (uniformly distributed within ±2 mm).

To sum up, for both scenarios (reference and modified parameter structures), the optimization was performed using three methods:

1. `lsqnonlin` (specified with analytical gradient)
2. `lsqnonlin` (unspecified; numerical gradient)
3. `patternsearch`

and each with two sets of calibration data: one noise-free, and one noisy. This leads to a total of 12 conditions, and each condition was tested five times for performance evaluation in terms of calibration speed and validation accuracy. Note that the initial parameters were kept the same for any of the six conditions within each scenario, and stopping criteria remained unchanged for all conditions; otherwise, the difference in results might also be attributed to different initialization and configuration.

The tests were run on a 2.9-GHz Intel Core i9 with 64 GB RAM and 20 CPUs using parallel computing (`parfor`): the processor was not overclocked, and the number of parallel workers was set as 14, which is a divisor of the total muscle number and equivalent to the maximal number of muscles to be simultaneously calibrated. The total time required for serial computing was calculated as the sum of individual calibration time for each muscle in parallel computing.

Results

For calibration based on noise-free data, the performance of the optimization methods is summarized in Table 1 and visualized in Fig 3, S1 Fig, Fig 4, and S2 Fig. With the reference parameter structure, the path calibration of all 42 muscles took 29.9 ± 1.4 min with our analytical gradient–based method, and in validation, only five moment arms contain error over 1 mm. Across all 182 moment arms, the average validation error is 0.12 mm and the max is 2.7 mm for the glenohumeral ab-/adduction moment arm of the long head of the biceps. When optimized with the numerically computed gradient, the total calibration time is 416.9 ± 12.2 min, and 42 moment arms across 13 muscles contain error more than 5 mm in validation. Muscles such as the serratus anterior superior and the scapular part of the deltoid contain errors over 10 mm in most of their moment arms. The performance of `patternsearch`, the non gradient–involved optimization, is overall similar to that of the numerical gradient–based method and no

match for our analytical gradient-based method, and occasionally absurd results were observed in validation (but excluded from comparison). Hence for clarity, the results of `patternsearch` will not be further discussed.

Table 1. Optimization performance (noise-free calibration data).

Test Condition		Calibration Speed		Validation Accuracy			
Method	Parameter Structure	Mean Computing Time (min)		Absolute Error (mm)		Moment Arm Count*	
		Serial	Parallel	Average*	Max*	Error > .5 cm	Error > 1 cm
`lsqnonlin` (anly.)	Reference	29.9 ± 1.4	3.7 ± 0.4	0.12	2.7	0	0
`lsqnonlin` (num.)	Reference	416.9 ± 12.2	59.8 ± 1.3	3.44	39.3	42	9
`patternsearch`	Reference	396.9 ± 5.6	53.0 ± 0.4	2.98	21.8	30	9
`lsqnonlin` (anly.)	Modified	50.4 ± 3.0	8.0 ± 0.7	0.51	7.7	2	0
`lsqnonlin` (num.)	Modified	297.4 ± 15.5	35.3 ± 0.9	3.27	43.0	33	9
`patternsearch`	Modified	297.7 ± 4.6	34.5 ± 0.3	2.61	21.7	19	4

*Calculated or counted from the 182 moment arms of all 42 muscles.

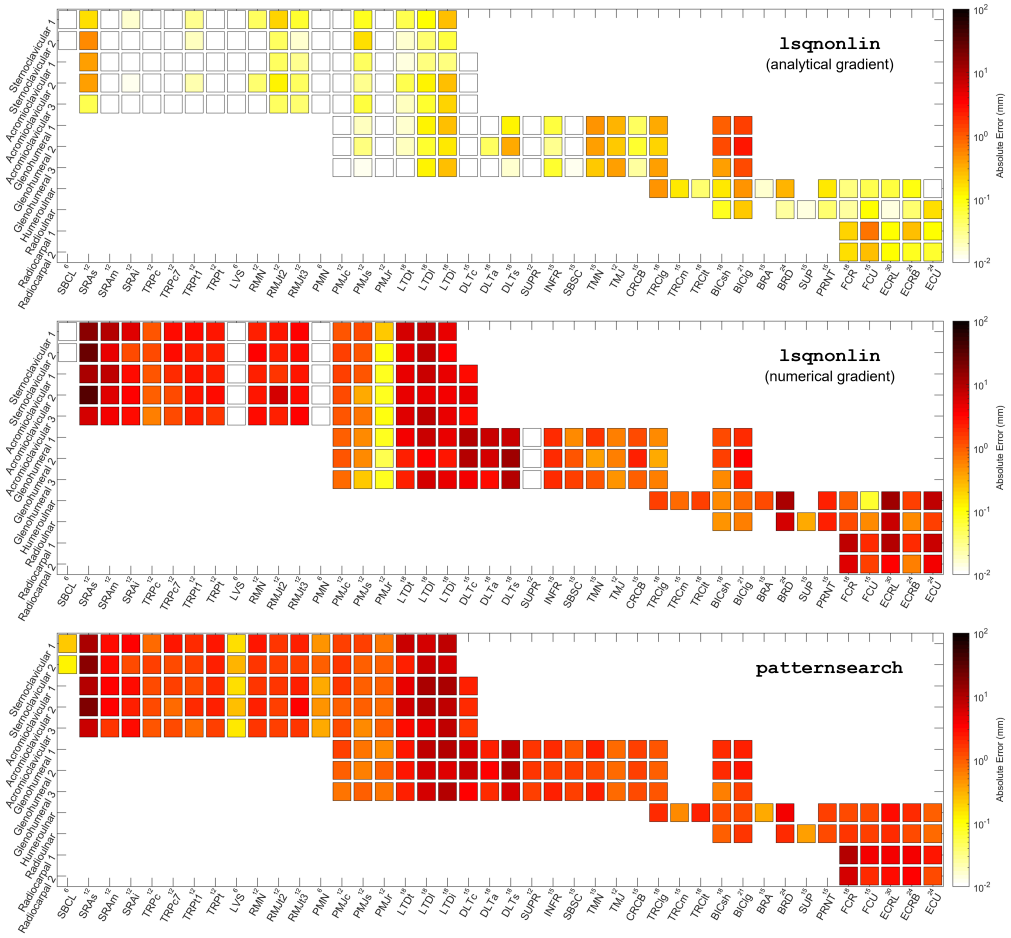


Fig 3. Absolute errors in 182 moment arms of 42 muscles in validation (reference parameter structure and noise-free calibration data). The magnitude of the error is indicated by the intensity of the color (color bar): the darker the color, the greater the error. The numbers in the horizontal axis correspond to the parameter numbers for each muscle. Notice the color bar is scaled logarithmically from 10^{-2} mm to 10^{2} mm. See S2 Table and S3 Table for nomenclatures in the axes.

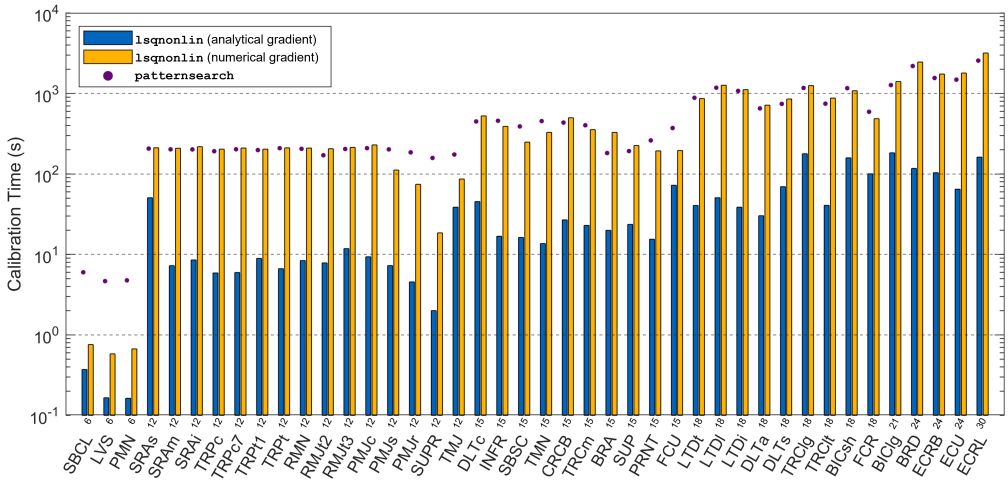


Fig 4. Individual calibration time for 42 muscles (reference parameter structure). The numbers in the horizontal axis correspond to the parameter numbers for each muscle. Notice the vertical axis is scaled logarithmically from 10^{-1} s to 10^{4} s. See S2 Table for the nomenclature in the horizontal axis.

With the modified parameter structure, the calibration time are 50.4 ± 3.0 and 297.4 ± 15.5 min respectively for the analytical and numerical gradient–based methods. In the former case, the average validation error is 0.51 mm, and only the clavicular and scapular parts of the deltoid contain errors over 5 mm in their glenohumeral ab-/adduction moment arms. In the latter case, the mean is 3.27 mm, and nine moment arms across five muscles contain more than 10 mm of error.

In Fig 4 and S2 Fig, it is worth mentioning that when calibrated without the gradient specified, most muscles with 12 parameters require about 3 min to calibrate, while most with 18 parameters require 16 min. The similarity in calibration time indicates that all starting points were iterated in the optimization, which is 300 sets for a 12-parameter muscle and 450 sets for 18-parameter. In other words, calibration error failed to reach the predetermined threshold even after complete iterations. This is rarely the case for gradient-specified calibration, where most muscles require much less time; calibration error per moment arm reached a sufficiently low level with only 10% or 40% of the iterations. However, there are a few exceptional cases (e.g., TMJ and FCU in Fig. 4, TMN and BRA in S2 Fig), where analytical specification did not improve optimization performance by a large extent. One possibility is that these muscles happened to be near-immobilized in the kinematics we selected to generate the calibration data, and the lack of variation in moment arms adds to the difficulty for optimization. Especially in the scenario of modified parameter structure, muscle paths are deprived of all via points set in the reference model (S2 Table), which inherently complicates the optimization, so it is not surprising that fewer muscles benefited from an analytical gradient.

The results for calibration based on noisy data are shown in S1 Table and visualized in S3 Fig, S5 Fig, S4 Fig, and S6 Fig. As can be expected, since the introduced noise is quite large, the advantage of our method is reduced. However, it is still faster and more accurate than optimization without gradient specification: About half the calibration time is required, while only a few muscles contain validation errors over 5 mm and almost none over 10 mm.

Figure 5 shows the visual resemblance in muscle path between the reference and calibrated models. Nevertheless, it is not without discrepancy; e.g., in a representative case, there is an apparent pointy bend in the upper arm (Fig. 5, yellow arrows). This is induced by the configuration of points rather than the optimization method: For the

lateral part of the triceps brachii, a via point is set on the posterior humerus to both imitate the bulge and prevent the path from penetrating into the bone. However, since this via point shares the same reference frame with the origin point, its location has no influence on the moment arm of humeroulnar flexion/extension, and the respective gradient is always 0. Consequently, the coordinates of this via point will not be optimized in the calibration of moment arm. For the same reason, the paths of the extensor carpi radialis longus and extensor carpi ulnaris are both distorted to some extent in the calibrated models. Most other muscle paths, especially in the shoulder, shared much resemblance to their reference.

Discussion

The key to optimization speed and accuracy is the gradient, which in our case is $\partial \boldsymbol{r}/\partial \boldsymbol{p}$, the gradient of the moment arm function with path-related parameters as variables. To derive it analytically, the cost function must be differentiable in the first place. For this, we revised the obstacle-set method by substituting its conditional statements and indifferentiable components with soft functions. Next, we disassembled the revised function into smaller modules that are easier to derive separately, and then assembled the gradients back into the desired gradient composite. The result is simplistic in form and thus easy to compute, which further increases optimization speed. Importantly, the concept of the chain rule is universal and may be useful for deriving the gradient of other complex functions as well.

The specification of gradient resolves most difficulties for the actual optimization, but not all of them. Ideally, had the optimization enough time to run with sufficiently small tolerances, there is not much need for any precondition or constraint. However, costs in computation and time are always a concern in reality, and we may as well include some conditions as long as the required information is not more than we already have.

To begin with, the type of obstacle as well as the number are predetermined in our method to define the vector structure of path parameters. It is of course possible to introduce a soft function that merges moment arm computation with different amount of various obstacles, so that the obstacle type and number can also be optimized. Yet we find it unnecessary at the current stage, since the cylinder is a more popular obstacle for muscle path modeling, and the obstacle number is highly related to the number of joints a muscle crosses so it can be easily predetermined.

We also included two constraints to solve the optimization problem: the locations of the origin and insertion points are limited to a range (± 10 mm) within the reference values, and the direction of wrapping is set the same in the reference model. The first constraint is not essential for our method, but it is also not necessary to optimize what is already quite clear. The origin and insertion points are anatomical landmarks, and their relative locations on the bones are well-studied. Especially if the goal is to calibrate muscle path for subject-specific models, the origin and insertion points should not shift more than our selected bounds, even with individual variability taken into account. The second constraint can be removed if a soft sign function is introduced, similar to the soft ReLU function included in our method. However, how muscle path wraps around the obstacle eventually depends on from which side the muscle crosses a joint, so this information is implied in the anatomy. Optimization without any precondition or constraint is indeed neat and tempting, but due to the practical reasons above, we introduced two conditions and two constraints for computational cost control.

Finally, there is the typical problem of local minimum, which cannot be avoided even with the optimization gradient and constraints. One approach is to rewrite Eq 1 into a regressor form, which is linear and has a lower risk of getting stuck in local minima, but

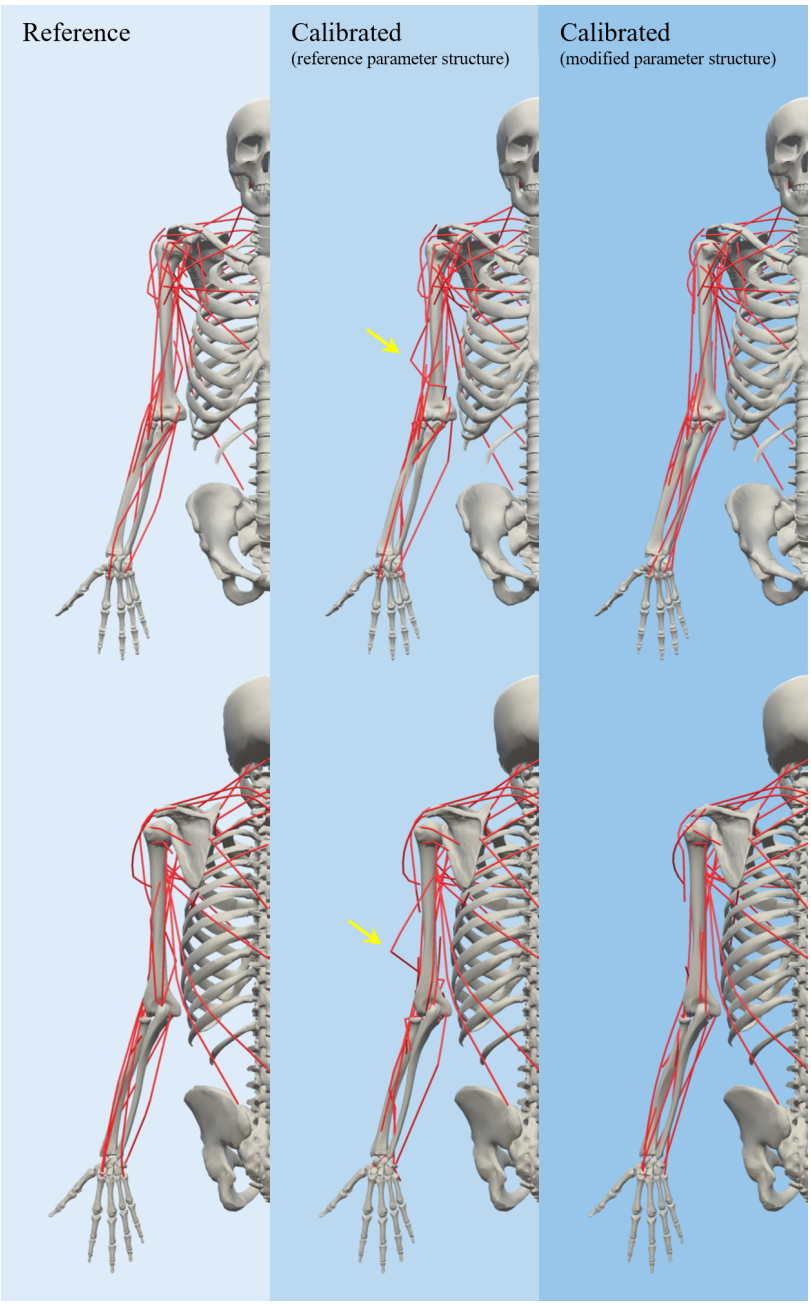


Fig 5. Front and back views of the reference and calibrated models. Calibration is performed with noise-free data and gradient-specified optimization. The yellow arrows point to the lateral part of the triceps brachii (TRClt). See S2 Table for the parameter structures in detail.

the derivation would be much more complicated. As an alternative, we adopted the common strategy of multiple starting points, and for each muscle, 150–750 different sets of random initial parameters are iterated in optimization. This does not eliminate the risk of local minimum, but should avoid it as much as possible. Again, it would be neat and tempting to hit the global minimum with only one shot, but our approach is more practical with the computational cost taken into account.

The performance of our method is demonstrated by replicating the muscle paths of a reference model based on the moment arm data generated by itself. Interestingly, even when the gradient is not specified, and with the established optimization frame alone, we have already achieved calibration with considerable speed and accuracy. Respectively in the case of the reference and modified parameter structures, 30 and 33 muscles do not contain any moment arm with more than 5 mm of error in validation. This level of accuracy is satisfactory enough for manual tuning, not to mention it takes only a few hours and may be further sped up using parallel computing.

When the gradient is specified in its analytical form, our method naturally outperforms optimization without specification, and we also demonstrated its advantage over pattern search. Nonetheless, we need to reiterate that muscle path calibration is challenging mainly because of the high dimensional relation between moment arm and joint configuration, which is often neglected. The moment arm in our reference model is not a single value but a vector of 12 elements. For instance, the pectoralis major crosses the shoulder complex and actuates up to seven DoFs, and each of the accordant moment arms varies with changes in any one of the seven DoFs, meaning that their moment arm–joint angle relation is depicted in a 7-D space. Calibrating the moment arms of the pectoralis major is similar to matching seven pairs of 7-D hypercubes, and it is impossible to accomplish manually unless four or five moment arms are neglected. In spite of that, our method succeeds in calibrating most muscles with great speed and accuracy: e.g., the three parts of the pectoralis major were each calibrated in a few seconds, and the validation errors in their 21 moment arms are all negligible.

One possible limitation of this study is that the performance of our method is demonstrated in silico with artificial data generated from a model. To eliminate the concern that our optimization performance mainly benefits from the same structure shared by the parameters in the reference model and those to be calibrated, we also tested the scenario when a simpler parameter structure is configured. As shown in Table 1, S1 Fig, and S2 Fig, though both calibration speed and validation accuracy decrease compared with when the reference parameter structure is configured, the results are still excellent. In reality, the path of a muscle is geometrically far more complicated than a few points and wrapping obstacles, but our results show that the optimization could still succeed with a relatively simple parameter structure. We also tested conditions with noisy data, and our method is quite robust and still hold significant advantages over the other two methods (S1 Table, S3 Fig, S4 Fig, S5 Fig, and S6 Fig). Importantly, our method is conveniently compatible with experimental data, since the only mandatory input is the relationship between moment arms and joint angles, regardless of how the data are obtained or whether the subject is human.

Another expected concern could be that, due to current technical limits, we usually do not have access to the measurements for all moment arms of a muscle, let alone their relations with all actuating DoFs. Yet this is not a problem related to our method but rather faced by all biomechanists. When lacking measurements, the moment arms will simply be assumed of certain values or relations, which is essentially what has been done in musculoskeletal modeling. Nevertheless, had such advanced measurements been available, our method is in line to provide equally advanced calibration. In fact, even if the high dimensional moment arm–joint angle relation is not completely established, our method is already usefully applicable. For example, to calibrate the Achilles tendon plantarflexion moment arm for a subject-specific model, one may use ultrasound or MRI to perform the measurement at the neutral ankle position, and then linearly scale the well-investigated generic moment arm curves [26–29] to obtain $\boldsymbol{r}_{\text{target}}(\boldsymbol{q})$ for our method. Or if one assumes some relation between moment arm and subject anthropometry, then they can also take moment arm values in the generic model as reference, scale them based on the assumed relation, and input in our method to generate a subject-specific

model. In either case, the resultant muscle path will be much more accurate than when path parameters are directly scaled based on skeletal geometry.

It is also worth noticing that, apart from the key role in muscle path calibration, our analytically derived gradient reveals conclusively the sensitivity of moment arm to path-related parameters. The value of each element in the gradient denotes how much influence each parameter has on moment arm in a certain joint configuration, which could be of important reference in medicine. For example, in the surgery repairing rotator cuff tears, the positional error of tendon reattachment or graft insertion would induce unwanted changes in moment arms, whose magnitudes may be different depending on the direction of positional shift. The information of sensitivity could help the surgeons or manufacturers of medical devices to focus on limiting the error in the direction with potentially larger moment arm variance, so as not to compromise the biomechanical function of the operated shoulder.

Finally, we recapitulate the universal practicability of the concept we employed in the derivation of optimization gradient. A complex function may be dissembled into a product of simple modules for separate derivation, and the gradients of each module not only can be re-assembled into the desired gradient composite, but may also be used as part of the gradient in other cost functions. A good example would be $\partial \boldsymbol{u}/\partial \boldsymbol{p}$ in Eq 14, from which we conveniently derived the gradient for a path geometry–based cost function. This enables calibration with medical imaging data (e.g., MRI), which not only would solve the visual discrepancy issue in this method, but could expand the possibility of application. Furthermore, it would be interesting if kinetic measurements such as joint moment can be utilized in calibration, which requires the optimization of musculotendon parameters. In a previous study, we have analytically derived the requisite gradient [12], and it can be integrated with this work, making possible joint moment–based model calibration. We are currently collecting relevant data and plan to perform a full-scale calibration in the future.

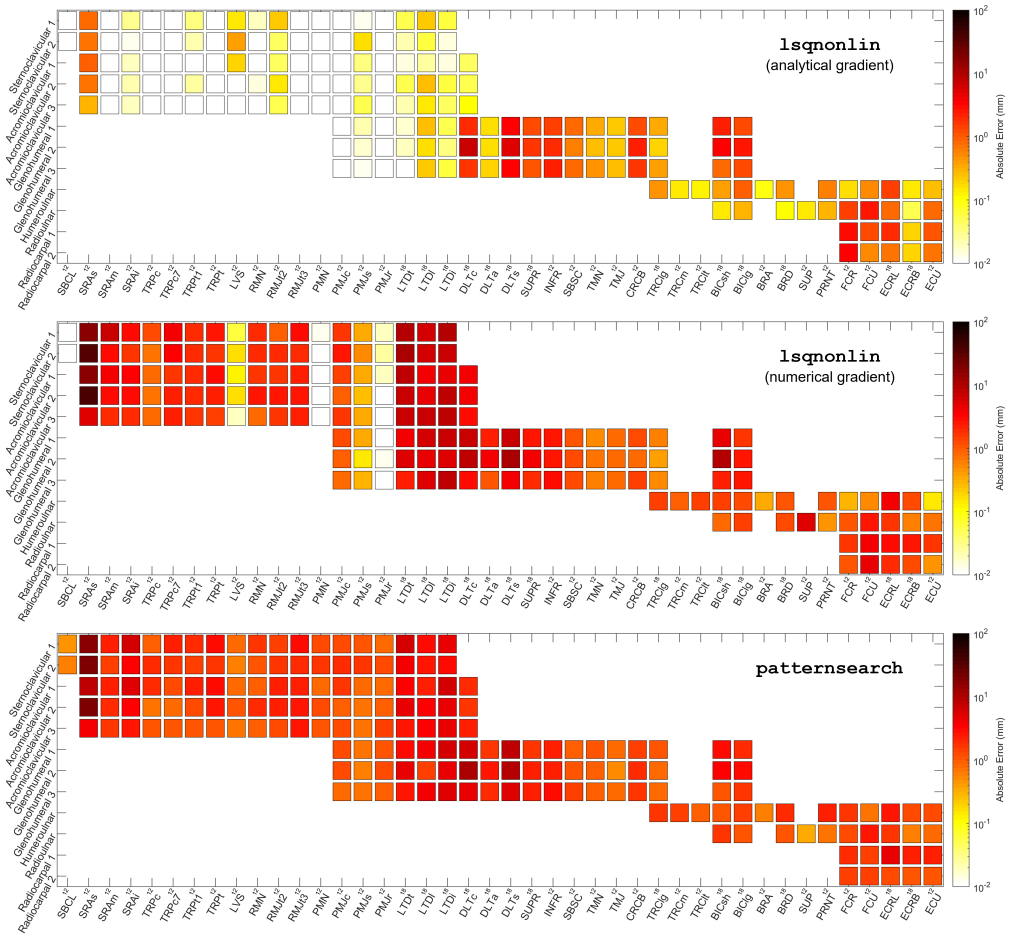
Conclusion

A gradient-based method is developed for automated muscle path calibration, where path-related parameters are optimized to minimize the error in muscle moment arm. This method features in the analytical form of the gradient of the moment arm function with path-related parameters as variables, and its performance is demonstrated by fast and accurate replication of paths from a 12-DoF 42-muscle human shoulder–arm model.

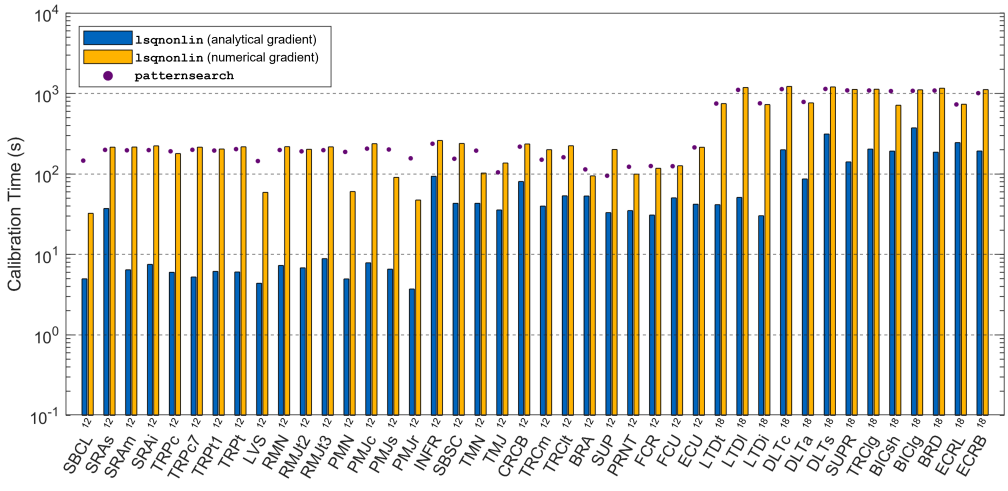
We explained the derivation in detail, which requires to first smooth the cost function by replacing the conditional statements and indifferentiable components with soft functions, and then modularize it into multiplicative factors that are easier to derive separately. This concept applies to other cost functions as well, and should be practical in the optimization of various properties in musculoskeletal models as well as many other systems.

Supporting information

S1 Fig. Absolute errors in 182 moment arms of 42 muscles in validation (modified parameter structure and noise-free calibration data). The magnitude of the error is indicated by the intensity of the color (color bar): the darker the color, the greater the error. The numbers in the horizontal axis correspond to the parameter numbers for each muscle. Notice the color bar is scaled logarithmically from 10^{-2} mm to 10^{2} mm. See S2 Table and S3 Table for nomenclatures in the axes.



S2 Fig. Individual calibration time for 42 muscles (modified parameter structure and noise-free calibration data). The numbers in the horizontal axis correspond to the parameter numbers for each muscle. Notice the vertical axis is scaled logarithmically from 10^{-1} s to 10^{4} s. See S2 Table for the nomenclature in the horizontal axis.

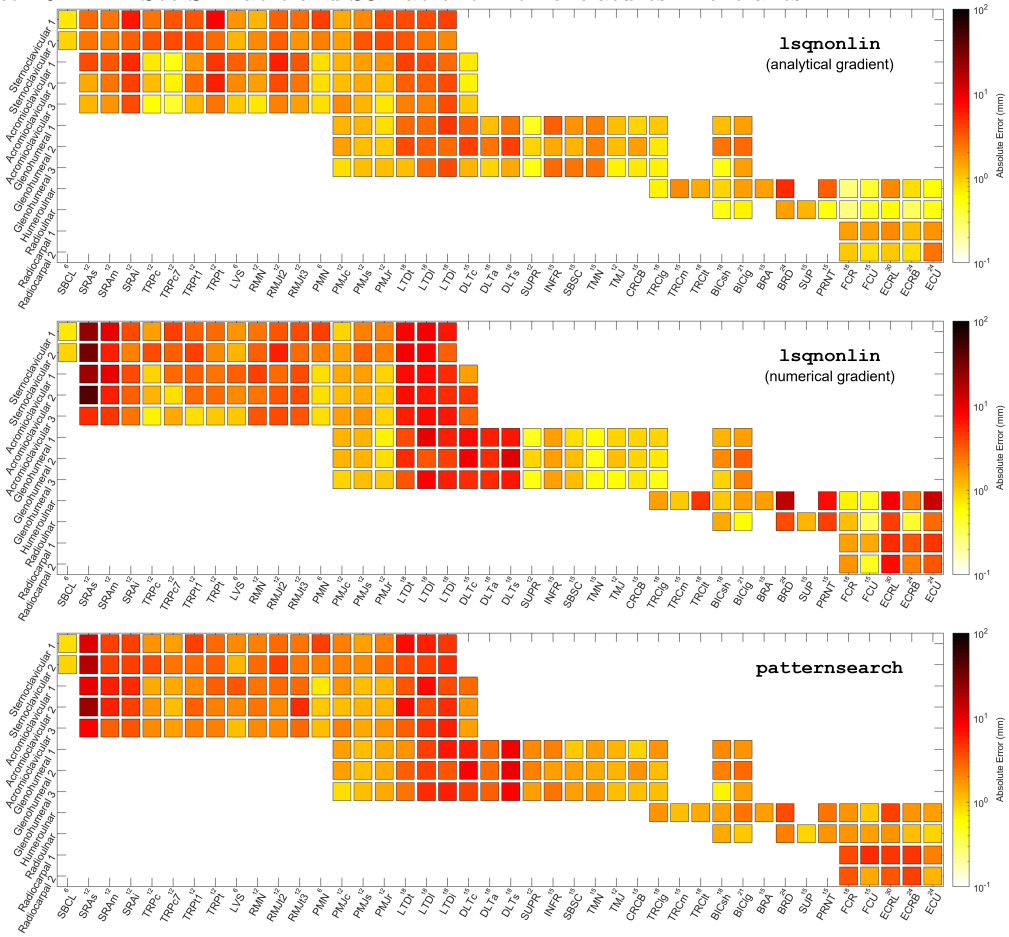


S1 Table. Optimization performance (noisy calibration data).

Test Condition		Calibration Speed		Validation Accuracy			
Method	Parameter Structure	Mean Computing Time (min) Serial	Parallel	Absolute Error (mm) Average*	Max*	Moment Arm Count* Error > .5 cm	Error > 1 cm
`lsqnonlin` (anly.)	Reference	145.0 ± 6.8	17.6 ± 0.6	2.16	8.3	6	0
`lsqnonlin` (num.)	Reference	466.6 ± 3.4	63.5 ± 0.3	4.01	48.3	44	10
`patternsearch`	Reference	406.6 ± 3.7	53.1 ± 0.1	3.13	21.8	27	5
`lsqnonlin` (anly.)	Modified	107.8 ± 2.9	12.2 ± 0.2	2.36	14.1	12	1
`lsqnonlin` (num.)	Modified	330.8 ± 0.9	37.5 ± 0.1	3.63	39.3	37	8
`patternsearch`	Modified	311.5 ± 3.2	34.8 ± 0.2	3.36	22.4	28	7

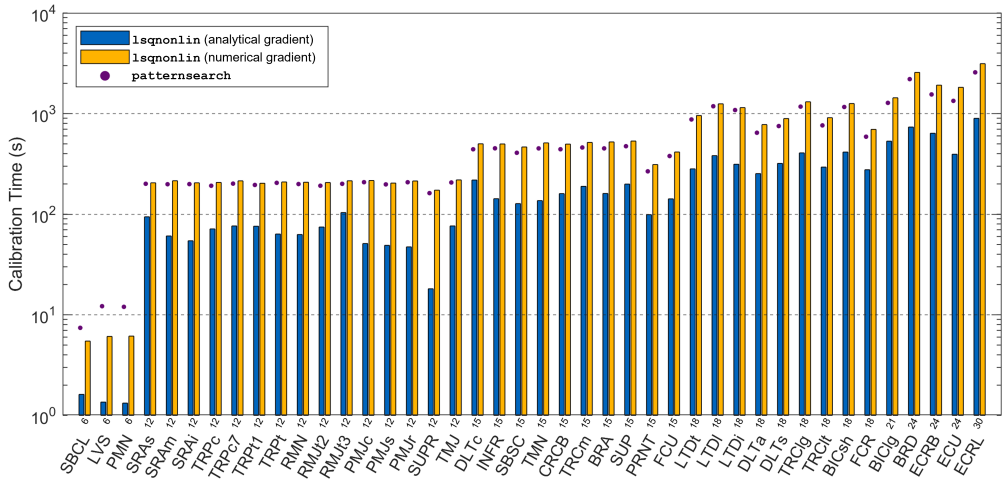
*Calculated or counted from the 182 moment arms of all 42 muscles.

S3 Fig. Absolute errors in 182 moment arms of 42 muscles in validation (reference parameter structure and noisy calibration data). The magnitude of the error is indicated by the intensity of the color (color bar): the darker the color, the greater the error. The numbers in the horizontal axis correspond to the parameter numbers for each muscle. Notice the color bar is scaled logarithmically from 10^{-1} mm to 10^{2} mm. See S2 Table and S3 Table for nomenclatures in the axes.

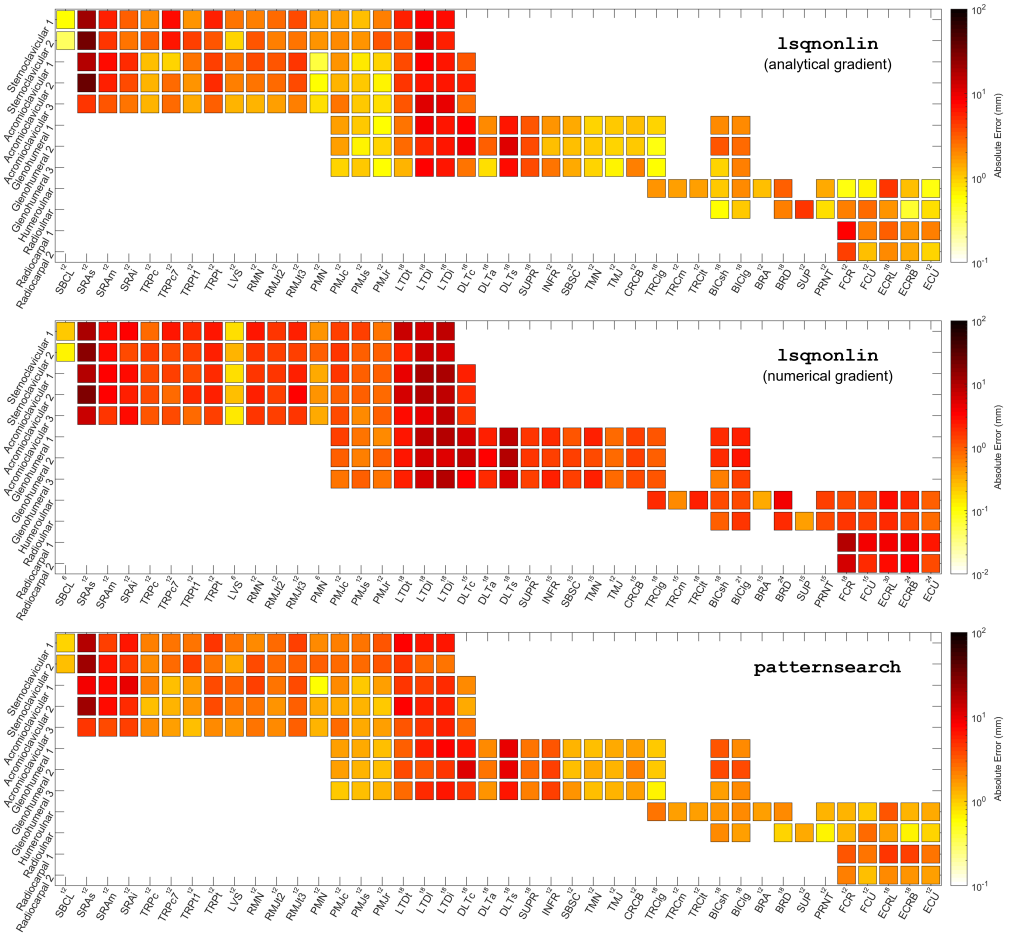


S4 Fig. Individual calibration time for 42 muscles (reference parameter structure and noisy calibration data). The numbers in the horizontal axis

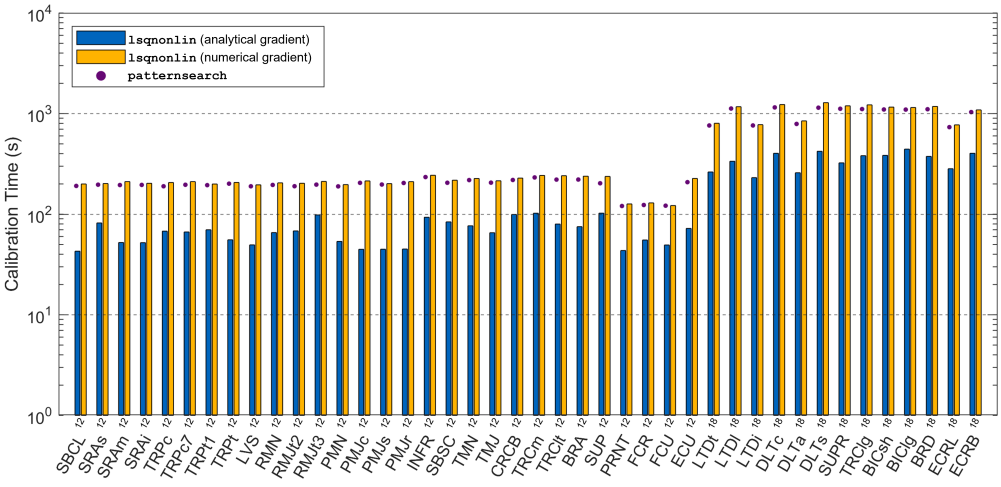
correspond to the parameter numbers for each muscle. Notice the vertical axis is scaled logarithmically from 10^0 s to 10^4 s. See S2 Table for the nomenclature in the horizontal axis.



S5 Fig. Absolute errors in 182 moment arms of 42 muscles in validation (modified parameter structure and noisy calibration data). The magnitude of the error is indicated by the intensity of the color (color bar): the darker the color, the greater the error. The numbers in the horizontal axis correspond to the parameter numbers for each muscle. Notice the color bar is scaled logarithmically from 10^{-1} mm to 10^2 mm. See S2 Table and S3 Table for nomenclatures in the axes.



S6 Fig. Individual calibration time for 42 muscles (modified parameter structure and noisy calibration data). The numbers in the horizontal axis correspond to the parameter numbers for each muscle. Notice the vertical axis is scaled logarithmically from 10^0 s to 10^4 s. See S2 Table for the nomenclature in the horizontal axis.



S2 Table. Nomenclature and path parameter structure for muscles in the human shoulder and arm. Parameter structure is denoted by a combination of the origin (O), via (V), and insertion (I) points, as well as cylinders (C) in the specified order. The number of parameters is shown in the bracket following each structure.

Abbreviation	**Muscle**	**Parameter Structure**	
		Reference	Modified
SBCL	subclavius	OI (6)	OCI (12)
SRAs	serratus anterior (superior)	OCI (12)	OCI (12)
SRAm	serratus anterior (middle)	OCI (12)	OCI (12)
SRAi	serratus anterior (inferior)	OCI (12)	OCI (12)
TRPc	trapezius (C1–C6)	OCI (12)	OCI (12)
TRPc7	trapezius (C7)	OCI (12)	OCI (12)
TRPt1	trapezius (T1)	OCI (12)	OCI (12)
TRPt	trapezius (T2–T7)	OCI (12)	OCI (12)
LVS	levator scapulae	OI (6)	OCI (12)
RMN	rhomboid minor	OCI (12)	OCI (12)
RMJt2	rhomboid major (T1–T2)	OCI (12)	OCI (12)
RMJt3	rhomboid major (T3–T4)	OCI (12)	OCI (12)
PMN	pectoralis minor	OI (6)	OCI (12)
PMJc	pectoralis major (clavicularis)	OCI (12)	OCI (12)
PMJs	pectoralis major (sternocostalis)	OCI (12)	OCI (12)
PMJr	pectoralis major (abdominalis)	OCI (12)	OCI (12)
LTDt	latissimus dorsi (thoracic)	OCCI (18)	OCCI (18)
LTDl	latissimus dorsi (lumbar)	OCCI (18)	OCCI (18)
LTDi	latissimus dorsi (iliac)	OCCI (18)	OCCI (18)
DLTc	deltoid (clavicular)	OCVI (15)	OCCI (18)
DLTa	deltoid (acrominal)	OVCVI (18)	OCCI (18)
DLTs	deltoid (scapular)	OVCVI (18)	OCCI (18)
SUPR	supraspinatus	OVVI (12)	OCCI (18)
INFR	infraspinatus	OVCI (15)	OCI (12)
SBSC	subscapularis	OVCI (15)	OCI (12)
TMN	teres minor	OVCI (15)	OCI (12)
TMJ	teres major	OCI (12)	OCI (12)
CRCB	coracobrachialis	OCVI (15)	OCI (12)
TRClg	triceps brachii (long head)	OCCI (18)	OCCI (18)
TRCm	triceps brachii (medial)	OVCI (15)	OCI (12)
TRClt	triceps brachii (lateral)	OVCVI (18)	OCI (12)
BICsh	biceps (short head)	OCCI (18)	OCCI (18)
BIClg	biceps (long head)	OCVCI (21)	OCCI (18)
BRA	brachialis	OVCI (15)	OCI (12)
BRD	brachioradialis	OVCCVI (24)	OCCI (18)
SUP	supinator	OCVI (15)	OCI (12)
PRNT	pronator teres	OVCI (15)	OCI (12)
FCR	flexor carpi radialis	OVCVI (18)	OCI (12)
FCU	flexor carpi ulnaris	OVCI (15)	OCI (12)
ECRL	extensor carpi radialis longus	OVCVVCVI (30)	OCCI (18)
ECRB	extensor carpi radialis brevis	OCVCVI (24)	OCCI (18)
ECU	extensor carpi ulnaris	OVVVCCI (24)	OCI (12)

S3 Table. Nomenclature for DoFs in the human shoulder and arm.

Sternoclavicular 1	Protraction/retraction
Sternoclavicular 2	Elevation/depression
Acromioclavicular 1	Lateral abduction/adduction
Acromioclavicular 2	Internal/external rotation
Acromioclavicular 3	Posterior abduction/adduction
Glenohumeral 1	Anteversion/retroversion
Glenohumeral 2	Abduction/adduction
Glenohumeral 3	Interior/exterior rotation
Humeroulnar	Flexion/extension
Radioulnar	Pronation/supnation
Radiocarpal 1	Palmarflexion/dorsiflexion
Radiocarpal 2	Radial/ulnar abduction

Acknowledgments

This work was supported by the Lighthouse Initiative Geriatronics by StMWi Bayern (Project X, grant no. 5140951) and the German Federal Ministry of Education and Research funding (Project AI.D, grant no. 16ME0539K).

References

1. Garner BA, Pandy MG. The obstacle-set method for representing muscle paths in musculoskeletal models. Computer Methods in Biomechanics and Biomedical Engineering. 2000;3(1):1–30. doi:10.1080/10255840008915251.

2. Hicks JL, Uchida TK, Seth A, Rajagopal A, Delp SL. Is my model good enough? Best practices for verification and validation of musculoskeletal models and simulations of movement. Journal of Biomechanical Engineering. 2015;137(2):020905. doi:10.1115/1.4029304.

3. Guo J, Huang H, Yu Y, Liang Z, Ambrósio J, Zhao Z, et al. Modeling muscle wrapping and mass flow using a mass-variable multibody formulation. Multibody System Dynamics. 2020;49(3):315–336. doi:10.1007/s11044-020-09733-1.

4. Lloyd JE, Roewer-Després F, Stavness I. Muscle path wrapping on arbitrary surfaces. IEEE Transactions on Biomedical Engineering. 2021;68(2):628–638. doi:10.1109/TBME.2020.3009922.

5. Delp SL. Surgery simulation: A computer graphics system to analyze and design musculoskeletal reconstructions of the lower limb [Doctoral dissertation]. Stanford University; 1990.

6. Holzbaur KRS, Murray WM, Delp SL. A model of the upper extremity for simulating musculoskeletal surgery and analyzing neuromuscular control. Annals of Biomedical Engineering. 2005;33(6):829–840. doi:10.1007/s10439-005-3320-7.

7. Arnold EM, Ward SR, Lieber RL, Delp SL. A model of the lower limb for analysis of human movement. Annals of Biomedical Engineering. 2010;38(2):269–279. doi:10.1007/s10439-009-9852-5.

8. Ma'touq J, Hu T, Haddadin S. A validated combined musculotendon path and muscle-joint kinematics model for the human hand. Computer Methods in Biomechanics and Biomedical Engineering. 2019;22(7):727–739. doi:10.1080/10255842.2019.1588256.

9. Gordon AM, Huxley AF, Julian FJ. The variation in isometric tension with sarcomere length in vertebrate muscle fibres. The Journal of Physiology. 1966;184(1):170–192. doi:10.1113/jphysiol.1966.sp007909.

10. Gollapudi SK, Lin DC. Experimental determination of sarcomere force–length relationship in type-I human skeletal muscle fibers. Journal of Biomechanics. 2009;42(13):2011–2016. doi:10.1016/j.jbiomech.2009.06.013.

11. Millard M, Uchida T, Seth A, Delp SL. Flexing computational muscle: Modeling and simulation of musculotendon dynamics. Journal of Biomechanical Engineering. 2013;135(2):021005. doi:10.1115/1.4023390.

12. Chen Z, Franklin DW. Musculotendon parameters in lower limb models: Simplifications, uncertainties, and muscle force estimation sensitivity. Annals of Biomedical Engineering. 2023;51(6):1147–1164. doi:10.1007/s10439-023-03166-5.

13. Weinstein JN, Andriacchi TP, Galante JO. A relationship of total knee design kinematics to patient function. The Iowa Orthopaedic Journal. 1987;7:85–90.

14. Aper RL, Saltzman CL, Brown TD. The effect of hallux sesamoid excision on the flexor hallucis longus moment arm. Clinical Orthopaedics and Related Research. 1996;325:209–217. doi:10.1097/00003086-199604000-00025.

15. Sherman MA, Seth A, Delp SL. What is a moment arm? Calculating muscle effectiveness in biomechanical models using generalized coordinates. In: 9th International Conference on Multibody Systems, Nonlinear Dynamics, and Control. American Society of Mechanical Engineers; 2013. p. V07BT10A052.

16. Arnold AS, Salinas S, Hakawa DJ, Delp SL. Accuracy of muscle moment arms estimated from MRI-based musculoskeletal models of the lower extremity. Computer Aided Surgery. 2000;5(2):108–119. doi:10.3109/10929080009148877.

17. Garner BA, Pandy MG. Musculoskeletal model of the upper limb based on the Visible Human Male dataset. Computer Methods in Biomechanics and Biomedical Engineering. 2001;4(2):93–126. doi:10.1080/10255840008908000.

18. Scheys L, Spaepen A, Suetens P, Jonkers I. Calculated moment-arm and muscle-tendon lengths during gait differ substantially using MR based versus rescaled generic lower-limb musculoskeletal models. Gait & Posture. 2008;28(4):640–648. doi:10.1016/j.gaitpost.2008.04.010.

19. Killen BA, Brito da Luz S, Lloyd DG, Carleton AD, Zhang J, Besier TF, et al. Automated creation and tuning of personalised muscle paths for OpenSim musculoskeletal models of the knee joint. Biomechanics and Modeling in Mechanobiology. 2021;20(2):521–533. doi:10.1007/s10237-020-01398-1.

20. Ao D, Li G, Shourijeh MS, Patten C, Fregly BJ. EMG-driven musculoskeletal model calibration with wrapping surface personalization. IEEE Transactions on Neural Systems and Rehabilitation Engineering. 2023;31:4235–4244. doi:10.1109/TNSRE.2023.3323516.

21. Delp SL, Anderson FC, Arnold AS, Loan P, Habib A, John CT, et al. OpenSim: Open-source software to create and analyze dynamic simulations of movement. IEEE Transactions on Biomedical Engineering. 2007;54(11):1940–1950. doi:10.1109/TBME.2007.901024.

22. McFarland DC, Binder-Markey BI, Nichols JA, Wohlman SJ, de Bruin M, Murray WM. A musculoskeletal model of the hand and wrist capable of simulating functional tasks. IEEE Transactions on Biomedical Engineering. 2023;70(5):1424–1435. doi:10.1109/TBME.2022.3217722.

23. Wretenberg P, Németh G, Lamontagne M, Lundin B. Passive knee muscle moment arms measured in vivo with MRI. Clinical Biomechanics. 1996;11(8):439–446. doi:10.1016/S0268-0033(96)00030-7.

24. McCullough MBA, Ringleb SI, Arai K, Kitaoka HB, Kaufman KR. Moment arms of the ankle throughout the range of motion in three planes. Foot & Ankle International. 2011;32(3):300–306. doi:10.3113/FAI.2011.0300.

25. Vaarbakken K, Steen H, Samuelsen G, Dahl HA, Leergaard TB, Nordsletten L, et al. Lengths of the external hip rotators in mobilized cadavers indicate the quadriceps coxa as a primary abductor and extensor of the flexed hip. Clinical Biomechanics. 2014;29(7):794–802. doi:10.1016/j.clinbiomech.2014.05.011.

26. Maganaris CN, Baltzopoulos V, Sargeant AJ. In vivo measurement-based estimations of the human Achilles tendon moment arm. European Journal of Applied Physiology. 2000;83(4-5):363–369. doi:10.1007/s004210000247.

27. Ward SR, Eng CM, Smallwood LH, Lieber RL. Are current measurements of lower extremity muscle architecture accurate? Clinical Orthopaedics and Related Research. 2009;467(4):1074–1082. doi:10.1007/s11999-008-0594-8.

28. Fath F, Blazevich AJ, Waugh CM, Miller SC, Korff T. Direct comparison of in vivo Achilles tendon moment arms obtained from ultrasound and MR scans. Journal of Applied Physiology. 2010;109(6):1644–1652. doi:10.1152/japplphysiol.00656.2010.

29. Holzer D, Paternoster FK, Hahn D, Siebert T, Seiberl W. Considerations on the human Achilles tendon moment arm for in vivo triceps surae muscle–tendon unit force estimates. Scientific Reports. 2020;10(1):19559. doi:10.1038/s41598-020-76625-x.

30. Hintermann B, Nigg BM, Sommer C. Foot movement and tendon excursion: An in vitro study. Foot & Ankle International. 1994;15(7):386–395. doi:10.1177/107110079401500708.

31. Lee SSM, Piazza SJ. Inversion–eversion moment arms of gastrocnemius and tibialis anterior measured in vivo. Journal of Biomechanics. 2008;41(16):3366–3370. doi:10.1016/j.jbiomech.2008.09.029.

32. Herzog W, ter Keurs HEDJ. A method for the determination of the force–length relation of selected in-vivo human skeletal muscles. Pflügers Archiv. 1988;411(6):637–641. doi:10.1007/BF00580859.

33. Delp SL, Hess WE, Hungerford DS, Jones LC. Variation of rotation moment arms with hip flexion. Journal of Biomechanics. 1999;32(5):493–501. doi:10.1016/S0021-9290(99)00032-9.

34. Wolfram S, Morse CI, Winwood KL, Hodson-Tole E, McEwan IM. Achilles tendon moment arm in humans is not affected by inversion/eversion of the foot: A short report. Royal Society Open Science. 2018;5(1):171358. doi:10.1098/rsos.171358.

35. Kane TR, Levinson DA. Dynamics: Theory and applications. McGraw-Hill; 1985.

36. Yamaguchi GT. Dynamic modeling of musculoskeletal motion: A vectorized approach for biomechanical analysis in three dimensions. Springer US; 2001.

37. Hu T, Kühn J, Haddadin S. Forward and inverse dynamics modeling of human shoulder–arm musculoskeletal system with scapulothoracic constraint. Computer Methods in Biomechanics and Biomedical Engineering. 2020;23(11):785–803. doi:10.1080/10255842.2020.1764945.

38. Hu T, Kuehn J, Haddadin S. Identification of human shoulder-arm kinematic and muscular synergies during daily-life manipulation tasks. In: 2018 7th IEEE International Conference on Biomedical Robotics and Biomechatronics. IEEE; 2018. p. 1011–1018.